International Journal of Computer Networks and Distributed Computing
Vol. 1, Issue 1

A Robuts and Fully Distributed Group Key Management

K. Kiran, Dr. D. Thilagavathy

Dept of CSE, Adhiyamaan College of Engineering, Hosur, T.N, India[1, 2]

ABSTRACT- An efficient and fully robust group key agreement protocol (GKA) enables a group of authenticated users to communicate over a reliable broadcast communication medium. All non-faulty nodes will form a cyclic group and hence have the same view of the message which is broadcasted and the faulty nodes cannot view that message. The standard encryption-based group key agreement protocol can be robust against an arbitrary number of node faults, because the performance deteriorates if some nodes fail during the protocol execution..By making each node to enter with the help of a nonce which is an arbitrary number that is used only once in a cryptographic communication and it will protect the malicious insiders that may disturb the group communication. The elliptic curve digital signature algorithm is used for establishing the group key and the proposed protocol has O(log n)-sized messages and expected round complexity close to 2, assuming random node fault and also it is secure under the (standard) Decisional Square Diffie-Hellman assumption.

Keywords: Group key agreement, fault-tolerance, security, Secure Communication, nonce.

I. INTRODUCTION

The growth of group applications triggers the need for group-oriented security mechanisms over insecure network channels. The applications include IP telephony, collaborative workspaces, secure conferences, as This all as dynamic coalitions common in law enforcement and disaster rescue scenarios. Standard security services required in such group settings, e.g. confidentiality of group-wide broadcasts, can be very efficiently achieved if all group members share a group- wide secret key. A group key agreement protocol (GKA) allows n players to create such shared secret key. There are several widely- known efficient constant-round group key agreement proto- cols [4, 8], but their performance degrades if some of the participating players fail during the protocol execution. This is a serious concern in practice, for example for mobile nodes that communicate over a wireless media, but which can loos connectivity during protocol execution.

Assuming a reliable broadcast medium, a GKA protocol can trivially be made robust to node failures by re-starting the protocol from scratch whenever a faulty player is detected. However, this would multiply all protocol costs by the number of faults, including the round complexity of the protocol. Robust constant-round GKA protocols can be achieved by executing parallel instances of any standard, i.e. non-robust, constant-round GKA protocol, one instance for every possible subset of non-faulty players. The early design of contributory group key agreement (GKA) protocols focuses on the efficiency of initial GKA. Efficiency metrics include computation, computation and round complexities. Although each metric is important in practice, the round complexity can be more crucial, particularly in the distributed computing environment.

Several well known efficient two-round GKA protocols are proposed in [12], [4]. However, their performance degrades if faults occur during the protocol execution. Faults cause the normal protocol (without robustness) to be restarted from the scratch. To improve performance, current GKA protocols must be made robust. In this context, robustness refers to the ability to complete the protocol, despite player and/or communication faults. Robust GKA is a serious concern in practice. Mobile nodes that communicate over a wireless medium can loos connectivity. Router failures, causing network partitioning (due to a mis-configuration or congestion) as malicious attacks, also increases the failure probability. List some motivating examples:

- Consider an emergent situation where some secure meeting for rescue missions and military negotiations must be held prior to a special time. In that case, robust GKA is prerequisite to minimize damage.

- Group communication (such as instant messaging and video- and audio- conferencing) operates on a real-time setting. Thus, robust GKA is crucial to improve the overall QoS.
- Security policies usually dictate that group keys must be refreshed periodically. Thus, a GKA protocol needs to be re-run (perhaps often), and improving GKA performance is essential.

- Consider a group of entities (routers or servers) in extreme environments, such as deep-space, that lack continuous network connectivity. In such a

setting, re-starting a GKA protocol, because a single participant failed, results in inordinately expensive costs.

Assuming a reliable broadcast medium, a GKA protocol can trivially be made robust to node failures by restarting the protocol from scratch, whenever a faulty player is detected. However, this would multiply all protocol costs by the number of faults, including the round complexity of the protocol. Robust constant-round GKA protocols can be achieved by executing parallel instances of any standard, i.e., non-robust, constant-round GKA protocol, one instance for every possible subset of non-faulty players. Such protocol would be robust and constant-round, but its communication and computation costs would grow by an inadmissible factor of 2n. Another robustness problem is caused by a malicious player, who sends arbitrary messages not correctly following the protocol. The goal of the adversary is to disrupt the protocol. One may think that message/player authentication can prohibit from sending random messages. However, authentication examines only authenticity of message/player, but does not determine if the player has sent the correct form of messages. In fact, well-known authenticated GKA protocols [3], [9] do not address the protocol disruption attack due to the malicious player.

II. SECURITY MODEL

Our security model is a standard model for Group Key Agreement protocols executed over authenticated links. Since the players in our GKA protocols do not use long-term secrets, This define GKA security.

A. Authenticated Links.

Our paper is concerned with Group Key Agreement (GKA) protocols in the authenticated links model. Note that there are standard and inexpensive compilation techniques which convert any group key agreement protocol into an authenticated group key agreement by (1) deriving a unique session-specific nonce at the beginning of the protocol and (2) having each player sign its message together with this nonce.

B. Broadcast Communication and Player Failure.

This assume that all communication within the protocol takes place over reliable (and authenticated) broadcast channel, where all the non-faulty players have the same view of the broadcasted message (which can be null if the sender is faulty). This assume weak synchrony, i.e., the players have synchronized clocks and execute the protocol in synchronized rounds, and the messages from the non-faulty players must arrive within some time window, which assume is large enough to accommodate clock skews and reasonable communication delays. The assumption of reliable broadcast communication might be realistic for certain communication scenarios, e.g. Ethernet or wireless communication between close-by players. Otherwise, reliable broadcast must be implemented via a consensus protocol.

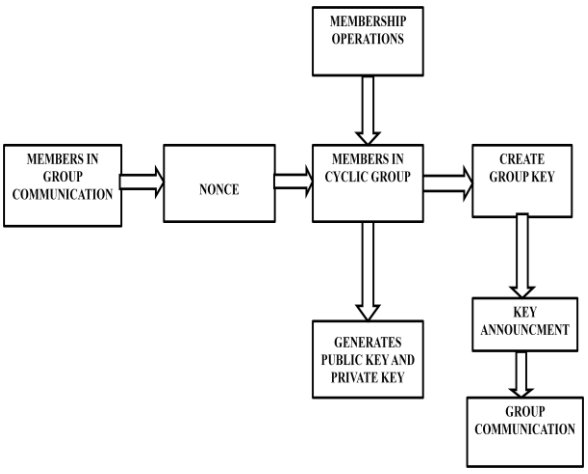
Assume an honest but curious adversary which can additionally impose arbitrary stop faults on the (otherwise honest) players participating in the protocol. Additionally, the adversary can make each player stop at an arbitrary moment in the protocol execution, but any such node failure cannot violate the contract imposed by the reliable broadcast assumption. Throughout the paper assume that these stop faults are scheduled in arbitrary way by the adversary, except in the last section when

Definition 1. (GKA Security) Consider an adversary algorithm A which observes an execution of the GKA protocol between n honest players, and, depending on bit b, is given the session key computed by this protocol (if b = 1) or a value chosen at random from the same domain as the sessions keys (if b = 0). The adversary A outputs a single bit b′. This define adversary's advantage in attacking the GKA protocol as:

$$GKA_A^{Adv} = | \Pr [b' = b] - 1/2 | \quad (1)$$

where the probability goes over the random execution of the protocol, the adversary A, and the random choice of bit b. This call a GKA protocol (ϙ, t)-secure if for all adversaries A who run in time t it holds that $GKA_A^{Adv} \leq$ €.

III. PROPOSED SYSTEM MODEL



IV. CRYPTOGRAPHIC SETTING

Let G be a cyclic group of prime order q, and let g be its generator. This assumes the DDH and Square-DDH problems are hard in G. For example, G could be a subgroup of order q in the group of modular residues Z^*p s.t. p − 1 divides q, |p| = 1024 and |q| = 160, or it can be a group of points on an elliptic curve with order q for |q| = 160.

Definition 2. The DDH problem is (€, t)-hard in G if for every algorithm A running in time t This have:

$$| \Pr[x, y \leftarrow Zq : A(g, gx, gy, gxy) = 1] - \Pr[x, y, z \leftarrow Zq : A(g, gx, gy, gz) = 1] | \leq € \quad (2)$$

Definition 3. The Square-DDH problem is (€, t)-hard in G if for every A running in time t have:

$$| \Pr[x \leftarrow Zq : A(g, gx, gx2\ ) = 1] - \Pr[x, z \leftarrow Zq : A(g, gx, gz) = 1] | \leq € \quad (3)$$

V. ROBUST GROUP KEY AGREEMENT PROTOCOLS

This describe our two-rounds robust GKA protocol that tolerates T faults with O(T)-sized messages, in three steps: In Sections 4.1 and 4.2, solely for presentation purposes, This explain how the non robust GKA protocol of Burmester-Desmedt (BD) [4] generalizes to a (fully) robust 2-round GKA protocol at the cost of increasing the length of the constant-sized messages of the BD protocol to O(n2)-sized messages. This call the robust generalization of the BD protocol BD-RGKA and show that the protocol remains secure under the same DDH assumption required for the underlying BD protocol. This show that the BD-RGKA protocol can be modified to retain full robustness with message size reduced to 2n group elements. This protocol remains secure under the same Square-DDH assumption, but its resilience is reduced to O(T) faults. (More precisely, the T-RGKA protocol tolerates all faults except two separate sequences of T or more consecutive faults.)

[Round 1]:
Each player P_i picks a random $t_i \in Z_q$ and broadcasts $z_i = g^{t_i}$.

[Round 2]:
Each P_i broadcasts its gadget value $X_{[i-1,i,i+1]} = (z_{i+1}/z_{i-1})^{t_i}$, where the indices are taken in a cycle.

[Key Computation]:
Each P_i computes the key as $\mathsf{sk}_i = (z_{i-1})^{nt_i} \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \cdots X_{i-2}$, where $X_i = X_{[i-1,i,i+1]}$.
(Note that for all i we have $\mathsf{sk}_i = g^{t_1t_2+t_2t_3+\ldots+t_nt_1}$.)

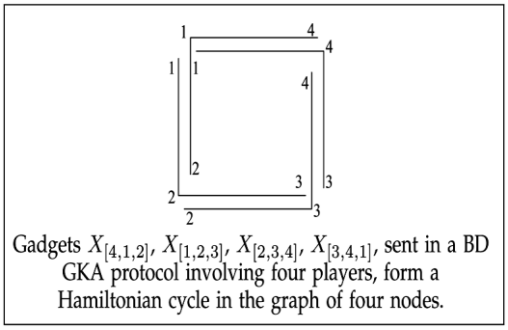
Fig.1. Burmester-Desmedt's Group Key Agreement Protocol (BD GKA).

Finally, This exemplify the usefulness of the efficiency versus fault-tolerance trade-off offered by the T-RGKA protocol, by showing that it implies a fully robust GKA protocol with 2 + δ expected rounds and messages of size O(log n + log(1/ δ), if the node faults are random and occur at a constant rate.

A. Overview: Adding Robustness to Burmester-Desmedt GKA

Since our fault-tolerant protocol is based on the GKA protocol proposed by Burmester and Desmedt (BD) [4], This need to first overview the BDGKA protocol to describe modifications This have made. The BDGKA protocol proceeds in two rounds (see Fig. 1): First, each player Pi broadcasts a public counterpart $z^i = g^{ti}$ of its contribution ti to the key.

In the second round, each Pi broadcasts X[i-1,I,i+1] = $g^{t1ti+1 = ti-1ti}$ (which it can compute as X [i-1,I,i+1] = (zi+1 = zi-1)^{ti}) . Given the set of values X[n,1,2],X[1,2,3] . . .;X[n-1;n,1], each player Pi can use its contribution ti to locally compute the common session key sk = $g^{t1t2+t2t3+\ldots+tnt1}$. This will call value X[i-1,I,i+1] a gadget, the titi+1 part of its exponent the left hand, and the ti_1ti part of the exponent, which is multiplied by minus one, the right hand of this gadget. A gadget X[i-1,I,i+1] corresponds to a path of length two connecting nodes Pi-1, Pi, and Pi+1. Using this graph terminology, This say that two gadgets are connectable if the left hand of one gadget is the same as the right hand of the other. For example, for every i, gadgets X[i-1,I,i+1] and X[i,i+1,i+2] are connectable. This say that a sequence of gadgets forms a path through the graph, if each two consecutive gadgets in the sequence are connectable. By inspecting the formula for deriving the secret key in the BD GKA protocol, This can observe that each player derives the same key because the set of gadgets broadcasted in the



Gadgets $X_{[4,1,2]}$, $X_{[1,2,3]}$, $X_{[2,3,4]}$, $X_{[3,4,1]}$, sent in a BD GKA protocol involving four players, form a Hamiltonian cycle in the graph of four nodes.

Fig.2. Gadgets in a BD GKA Protocol for n = 4.

second round of the protocol forms a Hamiltonian cycle (a.k.a. a "circular path") through the graph of all players. In Fig. 2, This show an example of four gadgets X [4,1,2], X[1,2,3], X[2,3,4], and X[3,4,1], created by the BD GKA protocol, executed in a group of four players.

B. Robust GKA with O(n^2) Message Size

This show the GKA protocol which follows the above idea, denoted BD-RGKA, in Fig. 3. The protocol is robust against any set of faults, and it remains secure under the same DDH assumption used by the basic BD GKA protocol. In other words, broadcasting all the additional information in the second round does not diminish the security of the protocol. Note that in the BD GKA protocol the session key sk = $g^{t1t2+t2t3+\ldots+tnt1}$ is computed according to a fixed circular order among the participating players, while in the BDRGKA protocol the session key is computed as sk = gta1ta2 + ta2ta3 +…+tamta1 , where Pa1 , . . . , Pam are players that remain alive after the second broadcast round. Note that, since This assume reliable broadcast and synchrony,

each player has the same view of the list of alive players and their messages. The alive players are ordered s.t. a1 < a2 <….. < am, but this order is arbitrary

[Round 1]:
1.1 Each P_i picks a random $t_i \in Z_q$ and broadcasts $z_i = g^{t_i}$.
[Round 2]:
2.1 Let AL be the list of indices of all players who complete Round 1.
2.2 Each P_i computes $X_{[k,i,j]} = (z_j/z_k)^{t_i}$ for all pairs (k, j) s.t. $k, j \in$ AL and $k \neq j$.
2.3 Each P_i broadcasts $\{X_{[k,i,j]}\}_{k,j \in \mathsf{AL}}$.
[Key Computation]:
3.1 Let AL be the list of indices of all players who complete Round 2.
3.2 Each P_i sorts the players in AL in the same order; wlog, we assume that the live players are ordered as $\{P_{a_1}, ..., P_{a_m}\}$, where $m \leq n$.
3.3 Each P_{a_i} computes the session key $\mathsf{sk}_{a_i} = (z_{a_{i-1}})^{m \cdot t_{a_i}} \cdot X_{a_i}^{m-1} \cdot X_{a_{i+1}}^{m-2} \cdot \cdots \cdot X_{a_{i-2}}$, where $X_{a_i} = X_{[a_{i-1}, a_i, a_{i+1}]}$.
(Note that $\mathsf{sk}_{a_i} = g^{t_{a_1} t_{a_2} + t_{a_2} t_{a_3} + \cdots + t_{a_m} t_{a_1}}$.)

Fig. 3.The BD-RGKA Protocol: Robust GKA with O(n2)- sized messages.

C. Fully Robust GKA Protocol with O(log n) Messages in the Random Fault Model

In this section, This show another robust GKA protocol, called RGKA', which is fully robust but not constant-round. RGKA' simply repeats the T-RGKA protocol above, with some parameter T, which This fix below, until T-RGKA succeeds. (In fact, only the second round of the T-RGKA protocol needs to be repeated, since the security of the BD-RGKA protocol implies that the ame contribution ti can be used in all these instances of the T-RGKA protocol.) Repeating the protocol increases the number of rounds and the protocol communication complexity. However, This will argue that if the faults are random and occur with rate _, then for any parameter _, the expected number of rounds in the RGKA' protocol can be 2 + δ , and the expected communication complexity per player can be 2(T + δ) group elements, for T[O((log n + log(1/δ))=log(1/v)). Assuming that the node faults are random and that they are independent of each other might seem unrealistic, but recall that the order among the participating players can be determined by the messages sent in the first round of the protocol, and, therefore, the usual dependencies between failures of nodes, which are physical neighbors, do not apply to the neighbors in the logical order created by the protocol.

This claim that if we set T ≈ (log n+ 1/2 log(1/ δ))/log(1/v) then the expected number of rounds in RGKA' is 2 + δ. Since a T-RGKA protocol succeeds exists except if at least two sequences of T consecutive nodes fail, the probability that a single execution of the T-RGKA protocol fails is upper-bounded by

$$f \leq n^2/2 * V^{2T}. \quad (4)$$

The expected number of rounds is then δ= 2(1/(1 – f)) -2 = 2f=(1 – f)≈2f. Therefore, by (1), This can upper-bound threshold T necessary to achieve parameters δ and V as T≤ (log n + 1/2 log(1/ δ))/ log(1/V).

D. Robust GKA with O(n) Message Size

Step 1: n2! 2n Reduction by Node-Doubling. The BDRGKA protocol achieves full robustness by increasing the message size by a factor of n2, but this overhead can be reduced to the factor of n as follows:

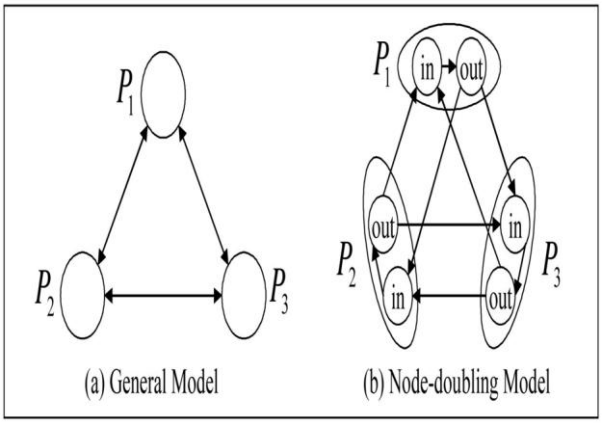


Fig. 4.Two different models of fully connected network for three players

Step 2: Reusing the Secret Contributions. This can reduce the message size of the above protocol by a factor of two, by having the two virtual nodes U2i-1 and U2i use the same secret contributions t2i-1 = t2i. This change implies that the gadgets created for the in nodes are inverses of the corresponding gadgets created for the out nodes.

[Round 1]: same as in *BD-RGKA* in Figure 3.
[Round 2]: same as in *BD-RGKA* in Figure 3, except:
2.2 Each P_i computes $X_{[k,i,i']} = (z_i/z_k)^{t_i}$ for all $k \in$ AL. Define $X_{[i,i',k]}$ as $(X_{[k,i,i']})^{-1}$.
[Key Computation]: same as in *BD-RGKA*, except:
3.3 Each P_{a_i} computes $\mathsf{sk}_{a_i} = (z_{a_{i-1}})^{m \cdot t_{a_i}} \cdot X_{a_i}^{m-1} \cdot X_{a_{i+1}}^{m-2} \cdot \cdots X_{a_{i-2}}$ as in the *BD-RGKA* protocol, but here X_{a_i} is defined as $X_{a_i} = X_{[a_{i-1}, a_i, a_{i'}]} \cdot X_{[a_i, a_{i'}, a_{i+1}]}$.
(Note that the resulting key is exactly as in *BD-RGKA* protocol because $X_{a_i} = g^{t_{a_i} t_{a_{i+1}} - t_{a_{i-1}} t_{a_i}}$.)

Fig. 5.The RGKA Protocol: Robust GKA with O(n)-sized messages.

E. Further Reduction of Message Size

In this section, This show two robust GKA protocols, TRGKA and RGKA'. The first protocol, T-RGKA, is the main

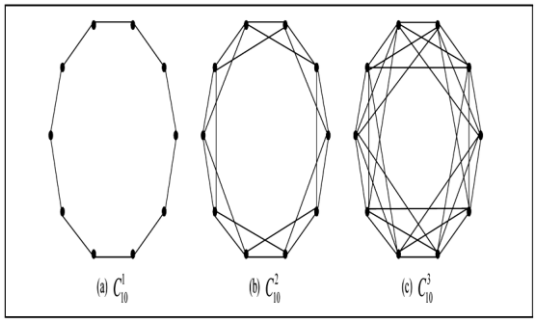


Fig.6. Examples of the T-th power of a circle.

VI. ROBUST GROUP KEY AGREEMENT EXTENSION

In this section, extend the robust GKA protocol to withstand the protocol disruption attack that the malicious player may attempt. While the basic robust GKA protocol considers missing gadgets (due to network or device failures), the extended robust GKA protocol additionally examines whether or not the gadgets generated by each player are consistent with the protocol algorithm. This refer to a gadget not correctly generated as a faulty gadget. Recalling that without a sequence of connectable gadgets, which covers the set of all nodes, the key agreement protocol fails, it is clear that a faulty gadget would lead to the protocol failure as well. However, the RGKA protocol can be still robust by excluding it, if a faulty gadget can be detected. In the following section, This illustrate a method to detect faulty gadgets and then construct two RGKA extensions by applying the method to the RGKA protocol. The detection method is also accordant with the T-RGKA, but This do not explain it because description on the RGKA protocol setting is enough.

VII. EFFICIENCY ANALYSIS

This first summarize the relevant aspects of protocol efficiency. Performance Criteria.

• Resilience: the number or pattern of faults that the protocol tolerates.

• Round Complexity: the number of rounds.

• Communication Complexity: the (expected) total bit-length of all messages sent in the protocol. (Since This assume a broadcast communication medium, This measure the bit-length of messages sent over a broadcast channel.)

• Computational Complexity: the amount of computation that must be performed per player in the protocol. This will restrict us to counting only the number of cryptographic operations (e.g. exponentiations and public-key operations) since these operations dominates the computational cost.

This compare the protocols This propose with non-robust BD protocol [4] and the encryption-based group key agreement protocol - the simplified version of CS protocol [5].Table 1 compares efficiency of the BD, the encryption-based GKA, denoted by "CS", and the BD-RGKA, RGKA, T-RGKA, and RGKA' protocols shown in the previous section. Of these six protocols, BD is not robust against even a single failure, T-RGKA is robust against at least 2T failures (see subsection

The conclusion we'd like to draw from this comparison is the following. First of all, all protocols run in two rounds, and RGKA' runs in expected $2+\delta$ rounds, for any δ, if T is set as $O(\log n+\log(1/\delta))$. (See Section 4.4.2 above for more discussion.) Given the comparable round complexities, the remaining important criterion is communication complexity. It is also computational complexity per player, but as the table shows, the latter follows the communication very

TABLE I
Complexity Comparison Between Provably Secure Protocols For Robust GKA Protocols

	Rounds	Communication	Computation
BD	2	2nt	3 ex
CS	2	(n + n2)t	2n pk
BD-RGKA	2	n3t	n2 ex
RGKA	2	n2t	n ex
T-RGKA	2	(1 + 2T)nt	(2 + 2T) ex
RGKA'	2 + δ	O(log n, log(1/δ))nt	O(log n) ex

VIII. CONCLUSION

In this paper, proposed a novel 2-round Group Key Agreement protocol that tolerates up to T node failures using (reliable) broadcasts of O(T)-sized messages. To authors' knowledge, it is the first GKA protocol that offers a natural trade-off between message size and the desired level of fault tolerance. In particular, This showed that the new protocol implies a fully-robust group key agreement with O(log n)-sized messages and expected round complexity close to 2, assuming random faults. The new protocol is secure under the (standard) Decisional Square Diffie-Hellman assumption.

ACKNOWLEDGMENT

Our thanks and wishes to the expects who contributed lot in the Group Key Agreement and This are trying to show the result in simulation model using NS-2.

REFERENCES

[1] Stanisław Jarecki, Jihye Kim, and Gene Tsudik "Flexible Robust Group Key agreement," IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 22, NO. 5, MAY 2011.

[2] Amir, Y., Nita-Rotaru, C., Schultz, J.L., Stanton, J.R., Kim, J.R., and Tsudik, G., "Exploring Robustness in Group Key Agreement," Proc.Int'l Conf. Distributed Computing Systems (ICDCS), pp. 399-408, 2001.

[3] Aditya, R., Peng, K., Boyd, C., Dawson, E., and Lee, B., "Batch Verification for Equality of Discrete Logarithms and Threshold

Decryptions," Proc. Second Int'l Conf. Applied Cryptography and Network Security (ACNS), pp. 494-508, 2004.

[4] Bresson, E., Chevassut, O., and Pointcheval, D., "Provably Authenticated Group Diffie-Hellman Key Exchange—the Dynamic Case," Proc. Conf. Asiacrypt '01, Dec. 2001.

[5] Burmester, M., and Desmedt, Y., "A Secure and Efficient Conference Key Distribution System (Extended Abstract)," Proc. Conf. Advances in Cryptology (EUROCRYPT '94), pp. 275-286, 1994.

[6] Boneh, D., "The Decision Diffie-Hellman Problem," Proc. Third Int'l Symp. Algorithmic Number Theory, pp. 48-63, 1998.

[7] Chaum, D., and Pedersen, T.P., "Wallet Databases with Observers,"Proc. 12th Ann. Int'l Cryptology Conf. Advances in Cryptology, pp. 89-105, 1992.

[8] Cachin, C., and Strobl, R., "Asynchronous Group Key Exchange with Failures," Proc. 23rd Ann. ACM Symp. Principles of Distributed Computing (PODC), pp. 357-366, 2004.

[9] Fiat, A., and Shamir, A., "How to Prove Yourself: Practical Solutions to Identification and Signature Problems," Proc. Conf. Advances in Cryptology (CRYPTO), pp. 186-194, 1986.

[10] Katz, J., and Yung, M., "Scalable Protocols for Authenticated Group Key Exchange," Proc. Conf. Advances in Cryptology (CRYPTO), pp. 110-125, 2003.

[11] Menezes, A., van Oorschot, P., and Vanstone, S., Handbook of Applied Cryptography. CRC Press, 1996.

[12] Schnorr, C.P., "Efficient Identification and Signatures for Smart Cards," Proc. Conf. Advances in Cryptology (CRYPTO), pp. 239-252, 1989.

[13] Steer, D.G., Strawczynski, L., Diffie, W., and Wiener, M.J., "A Secure Audio Teleconference System," Proc. Conf. Advances in Cryptology (CRYPTO), pp. 520-528, 1988.

[14] Steiner, M., Tsudik, G., and Waidner, M., "Key Agreement in Dynamic Peer Groups," IEEE Trans. Parallel and Distributed Systems, vol. 11, no. 8, pp. 769-780, 2000.